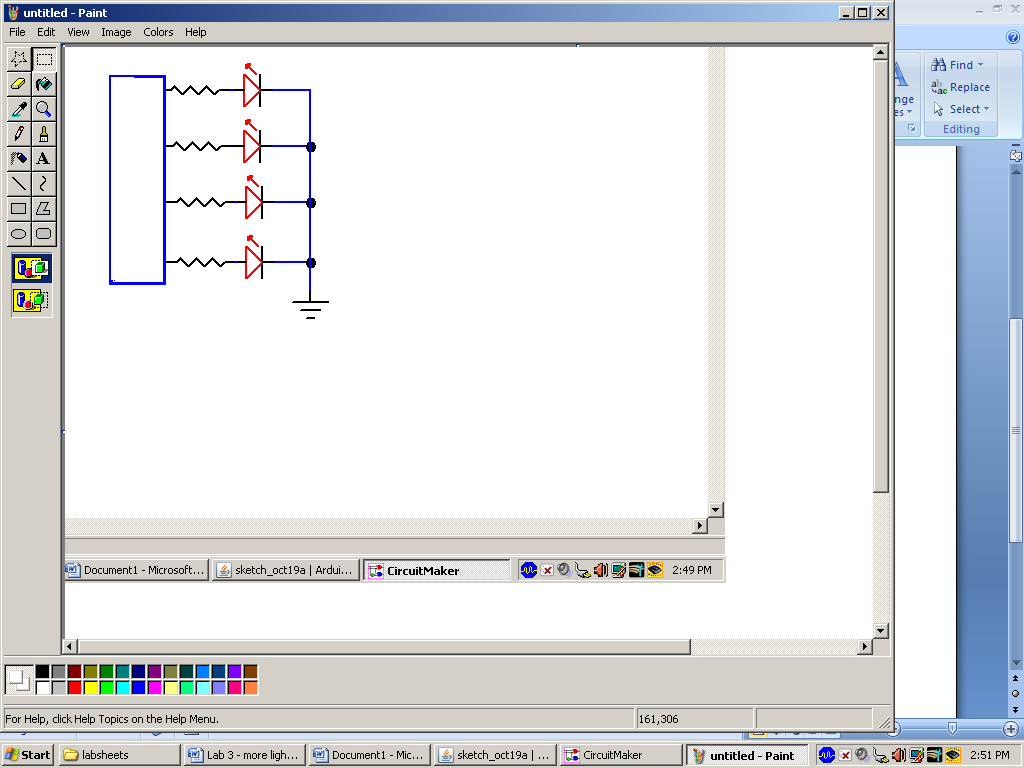
**Lab 3 – More Lights**

***The lab introduces controlling multiple outputs on the Arduino. You will learn how to use the breadboard to easily test circuits.***

**Hardware Required**Arduino Board LEDs (4 to start, 8 near the end)  
330 ohm resistors (4 to start, 8 near the end)  
miscellaneous jumper wires

**Circuit**

You need to know how a breadboard works before proceeding. Type “how to use a breadboard” in Google. The first hit, ([www.sciencebuddies.org/.../](http://www.sciencebuddies.org/.../Elec_) ) should provide you with some basic information about using the breadboard correctly. Read through the website before you attach four LED’s with resistors to the breadboard. Jumper wires connect the pins to the breadboard, run through the resistor and finally connect the LED’s to pins 3,4,5, and 6. Wire a common ground to simplify the connections. Obtain wiring help from your instructor, but do not connect the USB power cable to the Arduino without approval!

****

**Layout**

***At this point, you may not know, but the Arduino has many input and output pins. You will explore some of them with this lab. This lab is very similar to the first lab, but involves multiple outputs.***

**Basic Code Explanation**

In the program on page 2, the first thing you do is to “define” pin3 as an integer with the line: int pin3 = 3; There are other ways to define pin3, but use this one for now.

In the main loop, you turn the LED on with the line: digitalWrite(pin3, HIGH);

HIGH means “on”, and LOW means “off”.

This supplies 5 volts to pin 3. That creates a voltage difference across the pins of the LED, and lights it up.   
  
Then you turn it off with the line: digitalWrite(pin3, LOW);

This lab is very similar to the first lab, but involves multiple outputs.

Copy the code found on page 2

Sample code

------------------------------------------------------------------

*/\*  
  this program blinks four separate LED’s (one at time) and repeats  
\*/*

int pin3 = 3; // This defines “pin3” as a variable, much like

int pin4 = 4; // your math class would with “x” or “y”.

int pin5 = 5; // The variable *pin3* is actually an *integer*.

int pin6 = 6; // Here, each integer is assigned an Arduino pin.

// For now, accept that you need these to get your

// program working.

void setup() {

pinMode(pin3, OUTPUT); // initialize the digital pins(3-6)as outputs

pinMode(pin4, OUTPUT);

pinMode(pin5, OUTPUT);

pinMode(pin6, OUTPUT);

}

void loop() {

digitalWrite(pin3, HIGH); // set the LED connected to pin 3 on

delay(1000); // wait for a second (1000ms = 1 sec)

digitalWrite(pin3, LOW); // set the LED connected to pin 3 off

delay(1000); // wait for a second

digitalWrite(pin4, HIGH);

delay(1000);

digitalWrite(pin4, LOW);

delay(1000);

// Finish the program on your own

}

------------------------------------------------------------------

Once the program works, observe its behavior. Obtain the instructor’s initials here (\_\_\_)

For an added challenge (but actually a shortcut, change every “delay(1000)” to say “delay(tval)”  
and below int pin6=6, write “int tval = 1000;” . Run the program. It should stay the same. If you change the 1000 to another number, it changes every ‘tval’ to that new number. It saves you from changing each one individually. You can also have an ‘ontimeval’ and ‘offtimeval’ , meaning an integer for your time on, and an integer for time off. Give each variable a short, but descriptive name.

**Modify the Program**Program the LEDs to chase each other **quickly** (light up in succession). (\_\_\_)

Program the LEDs to all turn **on and off at the same time**. Instructor’s initials here (\_\_\_)

Program **eight** LEDs to “**bounce”** back and forth (similar to Knight Rider’s car, Kit). (\_\_\_)

Program your **own** and demonstrate it to your instructor. (\_\_\_)

Obtain two sets of LED’s (Red, Yellow, and Green) and create two traffic lights. On your breadboard, lay out an intersection. Search for light cycle timing (\_\_\_)